International Journal of Control, Automation, and Systems VV(X) (YYYY) 1-12
http://dx.doi.org/10.1007/s12555-xxx-xxxx-x
ISSN:1598-6446 eISSN:2005-4092
http://www.springer.com/12555

Synchronization of Networked Brachiating Robots

Praneel Acharya, Kim-Doang Nguyen*

Abstract: This paper investigates the motion synchronization of underactuated brachiating robots that communicate with each other over a network represented by a digraph. The synchronization is accomplished by leveraging the concept of a virtual leader, which leads to synchronization in stride time. For motion synchronization, three controllers are proposed: two model-dependent controllers and a model-free sliding-mode controller. All of the proposed controllers lead to a stable network operation. With a stable network, it is mathematically illustrated that all nodes synchronize their motion to the virtual leader. The performance of motion synchronization is evaluated quantitatively through measurements of errors and error rates for each physical robot in the network. Over time, a decrease in these metrics signifies improved synchronization and reduced deviation from the synchronized state. Simulation results demonstrate the efficacy of the controllers in achieving and maintaining synchronization across the network of physical robots. Model-based controllers can be limited because of the need to accurately know the system parameters for better performance. In this regard, we discuss a parameters estimation technique to approximate lumped parameters of the system.

Keywords: Bioinspired locomotion, Brachiation, Network control, Parameter estimation, Robust control, Underactuated robotics.

1. INTRODUCTION

The seamless integration of multiple robots into a single collaborative cyber-physical system to accomplish a shared goal has been an outstanding challenge in robotics that attracts a significant robotics research task force. Though substantial research efforts have focused on formation control [1–5], one equally interesting problem is motion synchronization. Accomplishing a cooperative task among multiple robots requires motion synchronization in many cases [6]. Early work on synchronization, such as the study of coupled oscillators detailed in [7], laid foundational groundwork for subsequent advancements in the field. Much of the research on network synchronization and consensus has traditionally emphasized the dynamics of agents with simple models, as noted in [8]. Noteworthy contributions include studies by [9–11], which focus on synchronization and consensus within linear systems. Recent developments have extended the scope to encompass weighted consensus to one-sided Lipschitz nonlinear fractional-order multiagent systems, as explored in [12]. While recent attention in controlled network synchronization has predominantly concentrated on Lagrangian systems, a significant portion of the literature has limited its focus to ***fully actuated*** systems [13, 14].

To fill this gap, this work targets the synchronization of networked ***underactuated*** brachiating robots. Underactuated brachaiting robots are class of underactaued euler lagrangian systems. Very few contributions have been produced to address motion synchronization of underactauted systems. For example, work in [15–18] discusses control strategy for multiple underactuated Lagrangian systems. However, these papers [16–18] considered underactuated dynamics that act as external disturbances to the system dynamics, i.e. manipulators mounted on an underactuated platform whose dynamics are influenced by uneven terrains or by waves in a high-sea state. In addition, work such as [19] are rooted in the fact that the interaction between neighboring oscillators is assumed to be anti-symmetric and proportional to the coupling strengths. Similarly, motion synchronization work in [20–22] does not consider different network topologies. Recent research focused on multilink single brachiating robots includes [23–26].

Since this work addresses the motion synchronization problem and incorporates brachiating and networked aspects, marking a notable departure from existing research primarily focused on single robot brachiation (see [23–26] for references). This study discusses synchronizing multiple brachiating robots capable of traveling in a networked setting. We have yet to find any other work that precisely matches these criteria. To the ***best of our knowledge***, this study on motion synchronization of networked brachiat-

Praneel Acharya is with the Department of Mechanical and Civil Engineering at Minnesota State University Mankato, MN, USA (e-mail: praneel.acharya@mnsu.edu).
Kim-Doang Nguyen is with the Department of Mechanical and Civil Engineering at Florida Institute of Technology, Melbourne, Florida, USA (e-mail: KNguyen@fit.edu).
* Corresponding author.

©ICROS, KIEE and Springer 2022

ing robots represents the first of its kind in this domain. In that regard, the key scientific contributions of this work are listed as follows:

- This is only work ***to best of our knowledge*** addressing the synchronization of networked, underactuated brachiating robots.
- The design of both model-dependent and model-free controllers is presented, and simulation results demonstrate their effectiveness.
- The proposed controller is distributed, requiring state information only from neighboring robots rather than the entire network.
- This work mathematically illustrates the controller's capability to synchronize the motion of any number of brachiating robots, scalable to any network size.
- Lumped parameter estimation using an iterative approach is discussed to approximate the physical properties of a brachiating robot.

The rest of the paper is organized as follows. Section 2 provides the background on how graphs can be used to describe the way different brachiating robots would communicate with each other. Section 3 describes two controllers that depend on the model parameters and Section 4 describes the model-free sliding mode controller. Section 5 proves the rigorous stability analysis of the proposed controllers. Section 6 illustrates that all nodes converge to the virtual leader achieving motion synchronization independent of how robots communicate with each other. Section 7 illustrates the simulation results for all the proposed controllers. Section 8 put forward an idea of lumped parameter estimation for model-dependent controllers. Finally, Section 9 provides several concluding remarks.

2. DYNAMIC BRACHIATION

2.1. Information flow

We consider a continuous-time multi-agent system with leader-follower dynamics. Leader dynamics differ from follower dynamics; only one leader exists, and any number of followers can exist. This work leader is a virtual oscillator, and followers are two link brachiating robots. Control inputs are applied to the robots; each robot can have different input values at a given time. The controller aims to achieve motion synchronization and brachiation of each robot with the leader. One way to synchronize the motion of robots and achieve brachiation is to learn how nature does it, which is to use a distributed method. In particular, each robot makes decisions based on the information received from its neighbor and how information flows among robots. This information flow can be described using directed graphs, as shown in Fig. 1, for example.

Information flow among multiple robots can be represented with a directed graph G with N nodes. Directed graph G is defined as $G = (V,E)$. Here, $V = \{v_1, v_2, v_3, \ldots v_n\}$ is the set of all the vertices or nodes where each vertex/node represents a robot and n is the total number of robots in the network. E_{ij} represents the edge connecting v_i node to another v_j node where $i \neq j$, describing the information flow between these nodes. Matrix $L = [l_{ij}] \in \mathbf{R}^{N\times N}$ which plays a similar role as the Laplacian matrix is defined as $l_{jj} = |\{E_{ij} \mid E_{ij} \in E \text{ and } E_{ij} \text{ ends at node } v_j\}|$, $l_{ji} = -1$ if there is an edge E_{ij} (starting at node v_i and ending at node v_j) and 0 otherwise. This enables the development of directed graphs with different topologies that dictate how information flows among robots.

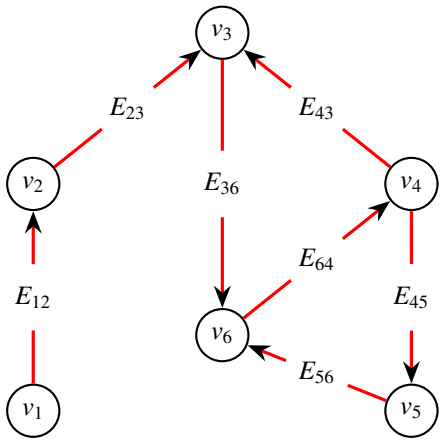

Fig. 1. Example of information flow in a directed graph.

In the case of a node connecting to multiple edges, information can be flowing from multiple edges to a node. For example (refer to Fig. 1), E_{23} and E_{43} indicate node v_3 receives information from nodes v_2 and v_4. Multiple edges E_{ij} can be used to completely define the information flow in a network. This enables the construction of a variety of weakly connected digraph network topologies, including ***directed acyclic graphs (DAGs)*** and ***digraphs with strongly connected components***, where all nodes except the leader node are reachable from the leader. With information flow being defined, we will now define the dynamics that govern each node in a network.

2.2. System Dynamics

Each node in the directed graph describes a brachiating robot (Fig. 2) whose dynamics are governed by the following underactuated equation of motion:

$$M_i(q_i)\ddot{q}_i + N_i(q_i,\dot{q}_i) = \begin{bmatrix} 0 \\ \tau_i \end{bmatrix}, \tag{1}$$

where i represents node v_i ($v_i \in V$), $q_i := [q_{1,i}, q_{2,i}]$ defined in Fig. 2, $M_i(q_i)$ is the mass inertia matrix for the robot in v_i node, and $N_i(q_i,\dot{q}_i)$ represents the nonlinearity, gravity, and disturbances for that node.

This paper focuses on the synchronization of stride time for all the brachiating robots in the network. Stride time is

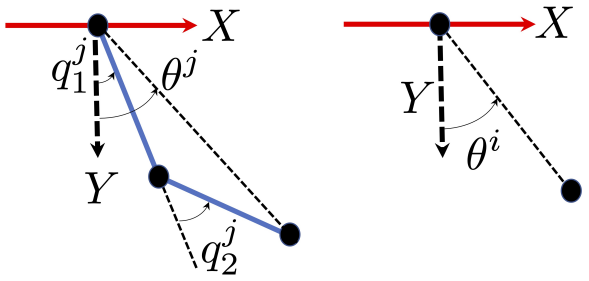

Fig. 2. Left: brachiating robot in v_j node; Right: virtual oscillator in v_i node.

defined as the time taken to complete a brachiation swing as the brachiating robot moves forward. To achieve synchronization, we use the concept of a virtual leader. The notion of a virtual leader to command a brachiator is inspired by the fact that the slow brachiation of apes can be approximated by the pendulum motion [27]. To be precise, the virtual leader is an oscillator governed by the following dynamics as discussed in [28].

$$\ddot{\theta}_j(t) + \omega^2 \theta_j(t) = 0, \tag{2}$$

where subscript j means the system belongs to the v_j node in the network. One important highlight is that in the case of motion synchronization, the virtual leader does not receive information about the state of other robots in the network.

3. MODEL-BASED CONTROLLER

With information flow defined in section 2 and dynamics for each robot defined in section 2.2, we develop several controllers independent of how information flows between the nodes. To do so, we look into each node v_i and the edges pointing to that node. The notion of considering a particular node v_i and edges pointing towards that node rather than the entire network topology fits with our motivation that v_i knows the state information of itself and its neighboring nodes that send their state measurements to v_i. Let S_i denote the set of robots that send their state measurements to the ith robot ($i \notin S_i$). Then, the desired trajectory for the ith robot is defined as:

$$\theta_{i_{\text{des}}} = \frac{(\theta_i + \sum_{j \in S_i} \theta_j)}{|S_i| + 1} = \frac{(\theta_i + r_i)}{p_i + 1} \tag{3}$$

For compactness of equations to be derived later on, $r_i = \sum_{j \in S_i} \theta_j$ and $p_i = |S_i|$, the size of S_i, are introduced. The desired trajectory (as shown in equation (3)) for each robot in the network is computed based on the state information that a robot receives from its neighboring robots. As an example, the desired trajectory for node v_6 as shown in Fig. 1 would be: $\theta_{6_{\text{des}}} = \theta_6 + \theta_3 + \theta_5)/(2+1)$. Thus, the node's desired trajectory is influenced by the state information of predecessor nodes. In this example, nodes v_5 and v_3 are predecessor nodes to nodes v_6. For a brachiator with equal link lengths, we have $\theta_i = q_{1,i} + 0.5 q_{2,i}$. For brachiating robots with different lengths, geometry can be used to express θ_i in the form of link lengths and q_i. Refer to [28] which explores the geometry of robots with unequal link lengths. All nodes in the network will employ the same controller as the rest of the nodes as a result we lay the foundation for ith node to track the desired trajectory $\theta_{i_{\text{des}}}$.

3.1. Partial feedback-linearization controller (Controller A):

To design a partial feedback linearization-based controller, we construct the following auxiliary variable for v_i node:

$$x_i = \dot{\theta}_i - \dot{\theta}_{i_{\text{des}}} + \lambda_{1,i}(\theta_i - \theta_{i_{des}}), \tag{4}$$

where $\lambda_{1,i}$ is a gain for the ith manipulator, $\theta_{i_{\text{des}}}$ is defined in (3), and $\dot{\theta}_i = d\theta_i/dt$. Now taking the time derivative of (4) and substituting (1) yields following:

$$\dot{x}_i = -p_i \left(\frac{2\bar{m}_{11,i} + \bar{m}_{12,i}}{2(p_i+1)} \right) n_{1,i} - \frac{\ddot{r}_i}{(p_i+1)} + \lambda_{1,i}(\dot{\theta}_i - \dot{\theta}_{i_{\text{des}}}) - p_i \left(\frac{\bar{m}_{22,i} + 2\bar{m}_{12,i}}{2(p_i+1)} \right) (n_{2,i} - \tau_i) \tag{5}$$

where $\bar{m}_{11,i}$, $\bar{m}_{12,i}$, $\bar{m}_{22,i}$ are the components of the inverse of the inertia matrix of the node v_i. Similarly, $n_{1,i}$ and $n_{2,i}$ are the vector components of the $N_i(q_i, \dot{q}_i)$. We pick the controller shown below for node v_i.

$$\tau_i = \frac{p_i(2\bar{m}_{11,i} + \bar{m}_{12,i})n_{1,i} - (2p_i\dot{q}_{1,i} - 2\dot{r}_i + p_i\dot{q}_{2,i})(\lambda_{12,i})}{p_i(2\bar{m}_{12,i} + \bar{m}_{22,i})} + \frac{p_i(2\bar{m}_{12,i} + \bar{m}_{22,i})n_{2,1} - (2q_{1,i} - 2r_i + q_{2,i})\lambda_{1,i}\lambda_{2,i} + 2\ddot{r}_i}{p_i(2\bar{m}_{12,i} + \bar{m}_{22,i})} \tag{6}$$

where $\lambda_{12,i} = (\lambda_{1,i} + \lambda_{2,i})$ and $\lambda_{2,i}$ is another positive gain associated with ith manipulator. Substituting the torque formulation (6) into (5) leads to $\dot{x}_i = -\lambda_{2,i} x_i$. Thus, torque ($\tau_i$) commanded to node v_i drags θ_i to converge exponentially to $\theta_{i_{des}}$. The convergence rate is influenced by the value of the positive number $\lambda_{2,i}$.

Acceleration data from neighboring nodes is essential to implementing the Controller A. In our simulation, when evaluating results for Controller A, each node stores its current acceleration to be passed to neighbors in the next iteration. This introduces a delay, but to minimize its effect, the simulation computes the new state every

0.001 seconds. This controller is ideal for real applications where the model information is known and acceleration can be accurately estimated. For instance, Kalman Differentiator method, discussed in [29] can estimate joint velocity and acceleration using only position measurements from an encoder. Additionally, acceleration can be approximated by differentiating velocity, with appropriate filtering to remove noise, as discussed in [30]. Furthermore, acceleration-based feedback has demonstrated feasibility and effectiveness in various robotic applications. For instance, in the work [31], an underactuated quadrotor control method utilizing direct acceleration feedback is proposed. Experimental tests demonstrate the controller's capability to achieve precise positioning under varying conditions. Similarly, research work in [32] employs acceleration feedback in parallel manipulators for trajectory tracking. So, the proposed controller is feasible in motion synchronization and brachiation of networked brachiating robots. However, we next propose two controllers that do not require acceleration information from neighboring nodes to compute the torque of the given node.

3.2. Partial feedback linearization with linear–quadratic regulator (Controller B)

Alternative to numerically computing the acceleration, we propose a new controller that relies on nodes transmitting their torque information, which is computed by each robot in every loop. To build this controller, we substitute $\ddot{q}_{1,i}$ and $\ddot{q}_{2,i}$ from (1) in $\ddot{\theta}_i = \ddot{q}_{1,i} + 0.5\ddot{q}_{2,i}$ which leads to the following equation:

$$\ddot{\theta}_i = \frac{-(2\bar{m}_{11,i} + \bar{m}_{12,i})n_{1,i} - (2\bar{m}_{12,i} + \bar{m}_{22,i})(n_{2,i} - \tau_i)}{2} \tag{7}$$

Now, we define the torque equation as follows:

$$\tau_i = u_{non,i} + \frac{2}{2\bar{m}_{12,i} + \bar{m}_{22,i}} u_{sync,i}, \tag{8}$$

where $u_{non,i} = \frac{(2\bar{m}_{11,i}+\bar{m}_{12,i})n_{1,i}+(2\bar{m}_{12,i}+\bar{m}_{22,i})n_{2,i}}{(2\bar{m}_{12,i}+\bar{m}_{22,i})}$. By substituting $u_{non,i}$ into the system dynamics equation, we obtain:

$$\ddot{\theta}_i = u_{sync,i}. \tag{9}$$

The above equation for each node can be represented using the state equation as follows:

$$\dot{y}_i = Ay_i + Bu_{sync,i}, \tag{10}$$

where $y_i = [\theta_i, \dot{\theta}_i]^T$, $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ and $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. With each node dynamics represented in the state-space form, the next step is to relate information flowing into the i^{th} node. To do so, we manipulate equation (10) to have two equations as listed below: one capturing the dynamics of node v_i and another capturing the dynamics of nodes sending the information to node v_i.

$$\frac{p_i}{(p_i+1)}\dot{y}_i = A\frac{p_i}{(p_i+1)}y_i + B\frac{p_i}{(p_i+1)}u_{sync,i}, \tag{11}$$

where, (11) is obtained by multiplying (10) by a constant $\frac{p_i}{(p_i+1)}$. Similarly, summing (11) associated with all nodes sending information to ith node leads to the following:

$$\frac{1}{(p_i+1)}\sum_{j\in S_i}\dot{y}_j = A\frac{1}{(p_i+1)}\sum_{j\in S_i}y_j + B\frac{1}{(p_i+1)}\sum_{j\in S_i}u_{sync,j} \tag{12}$$

Let $\bar{x}_i$ be an auxiliary variable defined as:

$$\bar{x}_i = \frac{p_i}{(p_i+1)}y_i - \frac{1}{(p_i+1)}\sum_{j\in S_i}y_j \tag{13}$$

The controller for node v_i is designed in the form listed below

$$u_{sync,i} = \Big(-K_i\bar{x}_i + \frac{1}{(p_i+1)}\sum_{j\in S_i}u_{sync,j}\Big)\frac{p_i+1}{p_i}, \tag{14}$$

where $K_i \in \mathcal{R}^2$ is the gain matrix. From (11), (12), and (14), we have the following auxiliary dynamics:

$$\dot{\bar{x}}_i = (A - BK_i)\bar{x}_i \tag{15}$$

The gain K_i can be designed such that $(A - BK_i)$ is a **negative definite matrix** which makes the auxiliary dynamics in (15) exponentially stable.

In implementation on real hardware, robot j calculates its control input and sends this value, along with its state, in a package to its neighbor i. Although there may be delays in the communication channels, these delays are typically very small unless the physical distances between robots are substantial, especially given today's communication and network technology. We assume that these communication delays are negligible and do not affect control stability.

The controllers designed so far (Controller A and Controller B) are dependent on the system parameters. As a result, the control performance depends on the accuracy of the dynamical model used. Uncertainties in system parameters and disturbances may affect the control performance. Therefore, a model-free controller is proposed in the next section for improved robustness.

4. MODEL FREE SLIDING MODE CONTROL

Designing the model-free sliding mode controller requires a sliding surface. Formulation (4) is used as a sliding surface, whose time derivative in (5) can be written in the form

$$\dot{x}_i = g_i\left(\frac{f_i}{g_i} + \tau_i\right), \tag{16}$$

where

$$\frac{f_i}{g_i} = \frac{p\,(2\bar{m}_{11,i} + \bar{m}_{12,i})\,n_{1,i} + p\,(\bar{m}_{22,i} + 2\bar{m}_{12,i})\,n_{2,1} + k_i}{-p(2\bar{m}_{12,i} + \bar{m}_{22,i})},$$

$$k_i = \frac{2\dot{r}_i - 2\lambda_{1,i}(\dot{\theta}_i - \dot{\theta}_{i_{des}})}{p(2\bar{m}_{12,i} + \bar{m}_{22,i})}.$$

In addition to the system being under-actuated, if the system is designed such that g_i is zero, then the control input will be meaningless. As a result, the brachiator is designed such that g_i is greater than zero.

The control input τ_i is designed based upon the bounded property of f_i/g_i and inspired by our previous work in [28]. Considering the brachiating robot as a rigid body: $\bar{m}_{11,i}$, $\bar{m}_{12,i}$, $\bar{m}_{22,i}$ are function of $\cos(q_{2,i})$ and its square. Nonlinear components $n_{1,i}$ and $n_{2,i}$ are function of $(\dot{q}_i)^2$ with coefficients as functions of $\sin(q_i)$. Thus, $\bar{m}_{11,i}$, $\bar{m}_{12,i}$, $\bar{m}_{22,i}, n_{1,i}, n_{2,i}$ are bounded. Besides the difference in the angles $(\theta_i - \theta_{i_{\text{des}}})$ is also bounded. Thus, the term f_i/g_i can be written as:

$$\left|\frac{f_i}{g_i}\right| < c_1 + c_2||\dot{q}_i||_2^2 \tag{17}$$

where, c_1, c_2 are positive numbers. With the above analysis, the controller is designed in the form,

$$\tau_i = -(c_0 + c_1 + c_2||\dot{q}_i||_2^2)\text{sat}(x_i/\varepsilon), \tag{18}$$

where ε is the width of a boundary layer and c_0, c_1, c_2 are the user-defined control parameter. Besides, the saturation function is defined as

$$\text{sat}(z_i) = \begin{cases} z_i, & |z_i| \le 1, \\ \text{sgn}(z_i), & |z_i| > 1. \end{cases} \tag{19}$$

Saturation function and bounding layer reduce the chattering problem associated with the sliding mode control.

We define the Lyapunov function candidate for a given node as

$$V_i = 0.5x_i^2 \Rightarrow \dot{V}_i = \dot{x}_i x_i = g_i\left(\frac{f_i}{g_i} + \tau_i\right)x_i. \tag{20}$$

When $|x_i| > \varepsilon$

$$\begin{aligned}\dot{V}_i &= g_i\left(\frac{f_i}{g_i} - (c_0 + c_1 + c_2||\dot{q}_i||_2^2)sat(\frac{x_i}{\varepsilon})\right)|x_i|sgn(x_i) \\ &\le -g_i|x_i|(c_0) < 0\end{aligned} \tag{21}$$

Thus, (21) guarantees when x_i is outside the boundary layer, it is driven towards the boundary layer. Once the system is inside the boundary layer, it is driven to the desired state governed by the first-order dynamics described in (4). Compared to Controller A and Controller B, Controller C does not use information about its neighbors' acceleration or torque.

5. ERROR DYNAMICS FOR THE NETWORK

So far three controllers have been proposed for a given node to track its desired trajectory. The goal of this section is to illustrate that if all nodes in the network implement any of the above-designed controllers, the stability of the networked system will be guaranteed, i.e. the error and error rates for all the physical brachiating robots decay over time. Out of (n) nodes, there are $(n-1)$ physical brachiating robots in the network. Node v_1 is the virtual leader throughout this work that does not take any input from any other nodes. Error for nodes except the virtual leader can be computed as follows:

$$e_i(t) = \theta_i(t) - \theta_{i_{\text{des}}}(t) \quad \forall i \in 2, 3, \ldots, n. \tag{22}$$

As a result, the error and error rates for all the physical nodes in the network can be represented as $e = [e_2, e_3, ..., e_n, \dot{e_2}, \dot{e_3}, ..., \dot{e_n}]^T$.

5.1. Error dynamics for networks with controller A:

Using equation (4) and $\dot{x}_i = -\lambda_{2,i}x_i$ following can be obtained:

$$\dot{e} = \begin{bmatrix} O_n & I_n \\ F_1 & F_2 \end{bmatrix} e = \xi e \tag{23}$$

Here, F_1 and F_2, defined as follows:

$$[F_{1,ij}] = \begin{cases} -\lambda_{1,i}\lambda_{2,i} & i = j \\ 0 & i \ne j \end{cases} \tag{24}$$

$$[F_{2,ij}] = \begin{cases} -(\lambda_{1,i} + \lambda_{2,i}) & i = j \\ 0 & i \ne j \end{cases} \tag{25}$$

It is obvious that both F_1 and F_2 are negative definite matrices.

Theorem 1: The error dynamics governed by (23) exponentially converge to zero.

Proof: Let $[\vec{a}_1, \vec{a}_2]^T$ be a eigenvector and the corresponding eigenvalue be α for the matrix ξ. Then:

$$\begin{bmatrix} O_n & I_n \\ F_1 & F_2 \end{bmatrix}\begin{bmatrix} \vec{a}_1 \\ \vec{a}_2 \end{bmatrix} = \alpha\begin{bmatrix} \vec{a}_1 \\ \vec{a}_2 \end{bmatrix} \Longleftrightarrow \begin{cases} \vec{a}_2 = \alpha\vec{a}_1 \\ F_1\vec{a}_1 + F_2\vec{a}_2 = \alpha\vec{a}_2 \end{cases} \tag{26}$$

Using substitution:

$$(F_1 + \alpha F_2)\vec{a}_1 = \alpha^2 \vec{a}_1 \tag{27}$$

Thus, $\vec{a}_1$ is also a eigenvector for $F_1 + \alpha F_2$ with the corresponding eigenvalue of α^2. Let P_s be the set containing all the eigenvalues of $F_1 + \alpha F_2$ where each eigenvalue in the set is represented by α_i^2 (for any i from 1 to $|P_s|$). If $\alpha \neq -\frac{\lambda_{1,i}\lambda_{2,i}}{\lambda_{1,i}+\lambda_{2,i}}$ then those eigenvalues in the set will be as follows:

$$\alpha_i^2 = -(\lambda_{1,i}\lambda_{2,i} + \alpha_i(\lambda_{1,i} + \lambda_{2,i})) \tag{28}$$

Solving for α_i we get the following:

$$\alpha_i = -\lambda_{1,i} \quad \text{or} \quad -\lambda_{2,i} \tag{29}$$

This implies the eigenvalues of ξ_1 are all negative. As a result, the network's collective error and error rates converge exponentially to zero when Controller A is applied. □

5.2. Error dynamics for networks with Controller B:

The same approach as discussed in section (5.1) can be used to show that the errors and error rates for the network when using Controller B exponentially converge to zero. In this case, F_1 and F_2 are defined as follows:

$$[F_{1,ij}] = \begin{cases} -K_{1,i} & i = j \\ 0 & i \neq j \end{cases} \tag{30}$$

$$[F_{2,ij}] = \begin{cases} -K_{2,i} & i = j \\ 0 & i \neq j \end{cases} \tag{31}$$

where $K_{1,i}$ and $K_{2,i}$ are the first and second entry of vector K_i. Thus, the conclusions established in section (5.1) are applied to the network with controller B. Next, we investigate the stability of the network when the model-free sliding mode controller is applied.

5.3. Error dynamics for the model-free sliding mode controller

The Lyapunov function candidate for the entire network can be written as follows:

$$V = V_2 + \ldots + V_n, \tag{32}$$

where, V_i is defined in (20). From (20), when $x_i > 0$ then $V_i > 0$ and from (21) when x_i is outside the boundary layer ε (i.e $|x_i| > \varepsilon$), we have $\dot{V}_i < 0$. The following conclusion can be made in light of (20) and (21).

- $V_i = 0$ if and only if $x_i = 0$.
- $V_i > 0$ if and only if $x_i \neq 0$.
- When $|x_i| > \varepsilon$ then $\dot{V}_i < 0$.
- $V = 0$ if and only if $[x_2, \ldots, x_n]^T = 0$.
- $V > 0$ if and only if $[x_2, \ldots, x_n]^T \neq 0$.
- When $\min_{2 \leq i \leq n} |x_i| > \varepsilon$ then $\dot{V}_2 + \ldots + \dot{V}_n < 0$.

Here, ε is the control parameter that a user can choose to define the boundary layer, i.e. the size of the neighborhood of zero that the system states converge to. In this work, we choose $\varepsilon = 0.001$. When $\min_{2 \leq i \leq n} |x_i| > \varepsilon$, all nodes are driven towards the neighborhood of zero due to the negative $\dot{V}$.

6. SYNCHRONIZATION WITH THE LEADER

To this point, it has been clearly shown that with **Controller A** and **Controller B**, the error converges to zero (a neighborhood of zero in the case of the sliding-mode controller) when times goes to infinity and represented as follows:

$$\lim_{t \to \infty} \left(\theta_i(t) - \theta_{i_{\text{des}}}(t) \right) = 0 \tag{33}$$

Substituting the definition of $\theta_{i_{\text{des}}}$ as presented in (3) in (33), we can get following equation:

$$\lim_{t \to \infty} \left(p_i \theta_i - \sum_{j \in S_i} \theta_j \right) = 0 \tag{34}$$

In light of this, the collective error for the entire network can be written as follows:

$$\lim_{t \to \infty} \underbrace{\begin{bmatrix} 0 & Z_{1\times(n-1)} \\ L_{2,(n-1)\times 1} & L_{1,(n-1)\times(n-1)} \end{bmatrix}}_{L} \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \tag{35}$$

where $\theta = [\theta_1, \theta_2, \ldots, \theta_n]^T$. $Z_{1\times(n-1)}$ is a zero matrix with 1 row and $(n-1)$ columns. We arrive at the following result in terms of leader-follower coordination.

Lemma 1: Physical nodes state θ_i in the network as expressed in (35) converge to θ_1 of the virtual leader.

Proof: Matrix L_1 is diagonally dominant with positive entries (p_i) along the diagonal. In light of the Gershgorin disk theorem [33], the eigenvalues of L_1 will be centered around the positive real part. Matrix L_1 is nonsingular as it has negative off-diagonal entries and its eigenvalues have real positive parts [34]. In addition, the property of L that the sum of entries along each row being zero can be mathematically written as $L_2 + L_1 \mathbf{1}_{n-1} = \mathbf{0}_{n-1}$, where $\mathbf{1}_{n-1}$ is the column vector of 1 and $\mathbf{0}_{n-1}$ is the column vector 0. This leads to $-L_1^{-1} L_2 = \mathbf{1}_{n-1}$.

It follows from (35) that

$$\begin{aligned} \lim_{t \to \infty} L_1 [\theta_2, \theta_3, \ldots, \theta_n]^\top &= -\lim_{t \to \infty} \theta_1 L_2 \\ \lim_{t \to \infty} [\theta_2, \theta_3, \ldots, \theta_n]^\top &= -\lim_{t \to \infty} \theta_1 (L_1^{-1} L_2) \end{aligned} \tag{36}$$

Therefore we have:

$$\lim_{t\to\infty}[\theta_2,\theta_3,...,\theta_n]^T = \lim_{t\to\infty}\theta_1 \tag{37}$$

Equation (37) indicates that all the nodes in the network have their state θ_i converged to the virtual leader θ_1 as time goes to infinity. Despite introducing the concept of a virtual leader, any node can act as a leader, and all other nodes synchronize their motion to that leader using the proposed controllers. This study emphasizes the concept of a virtual leader, ensuring the leader's performance remains unaffected by external factors. This raises the pertinent question: What happens without a virtual leader? Upon investigation without a virtual leader (a leaderless network), all nodes were observed to synchronize their motion; however, brachiation did not occur. This observation is supported by [35], which states, "In the absence of a leader, networked agents, when stable, often settle into a consensus state." A consensus state is usually a constant value. However, achieving brachiation, particularly the swinging motion, requires a leader to guide and coordinate the motion effectively. This highlights the critical role of a leader that initiates the brachiation motion. In the next section, we implement the proposed controllers for multiple nodes communicating with each other to illustrate the motion synchronization of followers with the virtual leader.

□

7. SIMULATION RESULTS

In the simulations, we select the brachiating robots with properties shown in Table 1. All controllers are tested with two different networks describing the information flow as shown in Fig. 3 and Fig. 4. In the network in Fig. 3, each node receives information from one node and follows a tree-like structure. In the network in Fig. 4, the nodes receive information from multiple nodes and follow a loop-like structure. In these examples, node v_1 represents the virtual leader which is an oscillator and the other nodes represent the brachiating robots. Additionally, all brachiating robots have the same geometric properties. The initial conditions of the virtual leader are as follows: $\theta(0) = -\frac{\pi}{2}$ rad, $\dot{\theta}(0) = 0.0$ rad/sec and $\omega = 3.4$ Hz. The initial conditions for different nodes as described by different networks are listed in Table 2 and 3, respectively.

Table 1. Brachiating robot physical properties.

	mass	length	inertia	center of gravity
link 1	3.499	0.5	0.090	0.414
link 2	1.232	0.5	0.033	0.333

Table 2. Randomly generated initial condition for robots with information flow represented in Fig. 3.

	q_1 (deg)	q_2 (deg)	$\dot{q}_1$ (deg)	$\dot{q}_2$ (deg)
v_2	25.1238	30.5624	-0.8959	0.7188
v_3	18.6118	-40.4866	-1.524	-0.0065
v_4	55.1693	-19.1537	0.3411	-1.1048
v_5	30.152	-29.3886	0.0238	0.7963
v_6	46.9084	55.115	0.1889	-1.4455
v_7	-42.0847	-29.099	1.3629	-0.9829
v_8	37.7142	-30.777	1.7171	-0.6001
v_9	-36.4086	-29.8699	0.4642	-0.1068
v_{10}	-17.8009	39.6994	0.3411	0.1989

Table 3. Randomly generated initial condition for robots with information flow represented in Fig. 4.

	q_1 (deg)	q_2 (deg)	$\dot{q}_1$ (deg)	$\dot{q}_2$ (deg)
v_2	23.3794	-21.9481	1.8009	-1.8622
v_3	-7.3507	-14.213	1.0621	1.1808
v_4	-37.5753	-1.2283	-0.2177	0.5853
v_5	25.1238	30.5624	-0.8959	0.7188
v_6	18.6118	-40.4866	-1.524	-0.0065
v_7	55.1693	-19.1537	0.3411	-1.1048
v_8	30.152	-29.3886	0.0238	0.7963
v_9	46.9084	55.115	0.1889	-1.4455
v_{10}	-42.0847	-29.099	1.3629	-0.9829

7.0.1 Controller A results

The control parameters values $(\lambda_{1,i}, \lambda_{2,i})$ discussed in Section 3.1 are selected to be $\lambda_{1,i} = 6.5$ and $\lambda_{1,i} = 7$ for all nodes. In addition, these gain values are kept fixed for two different networks illustrated in Fig. 3 and Fig. 4. The result of applying controller to two networks is illustrated by Figs. 5 and 6.

As seen in Fig. 5, all nodes connected as in Fig. 3 synchronize their motion with respect to the virtual leader v_1 within 1 second of the motion despite nodes starting from different initial condition. Similar results are obtained in Fig. 6 where nodes are communicating with each other as described in Fig. 4. After every stride, though the states are disturbed by the switching of vector fields and impact, they again quickly synchronize their motion with the leader.

7.0.2 Controller B results

The controller is designed in (8). The gain values for all nodes are selected as $K_i = [121, 22]^T$. The simulation result is shown in the Fig. 7, which shows the nodes' motion synchronized after 3 seconds, equivalent to three strides. Similar observations were observed for the network in Fig. 4 with gain values of $K_i = [1000, 205]^T$.

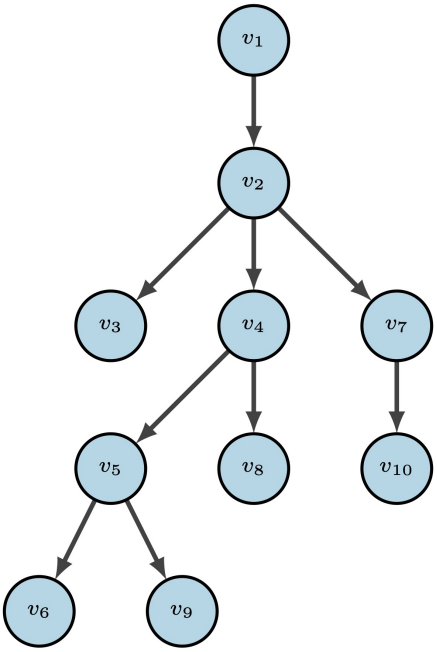

Fig. 3. Tree-like network connection with node v_1 being the virtual leader and other physical brachiating robots.

7.0.3 Results for the model-free sliding mode controller

The controller is formulated in (18). We select $c_0 + c_1 = 5$ and $c_2 = 0.9$ for the network in Fig. 3. The results are shown in Fig. 9, where all nine robots synchronize their motion within 4 seconds. Similar results are observed in Fig. 10 for the network topology in Fig. 4.

Up to this point, we have discussed three different controllers, ranging from model-dependent to model-free, which have been demonstrated to achieve synchronized motion and brachiation with a virtual leader within a specified network. As this is the only work we are aware of that addresses networked brachiating robots, we currently lack direct comparisons with other methods. To facilitate such comparisons, we are exploring literature discussing control strategies for single brachiating robots rather than networks of robots. One relevant study is presented in [28], where a control algorithm is proposed for the brachiation of a single robot. The proposed controller achieves brachiation within 1 second for a single brachiating robot starting from a fixed initial condition. Simulation results in Fig. 5 and 6 demonstrate that controller A, defined in equation (6), enables all nine robots to brachiate within one second, unlike previous methods that achieve this for only a single robot. These nine robots begin from various random initial conditions and are interconnected based on different network graphs. Therefore, controller A accomplishes both motion synchronization and brachiation within one second rather than just achieving brachiation

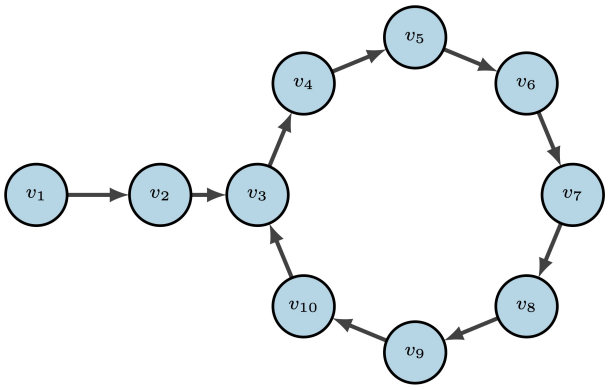

Fig. 4. Cyclic network connection with node v_1 being the virtual leader and other physical brachiating robots.

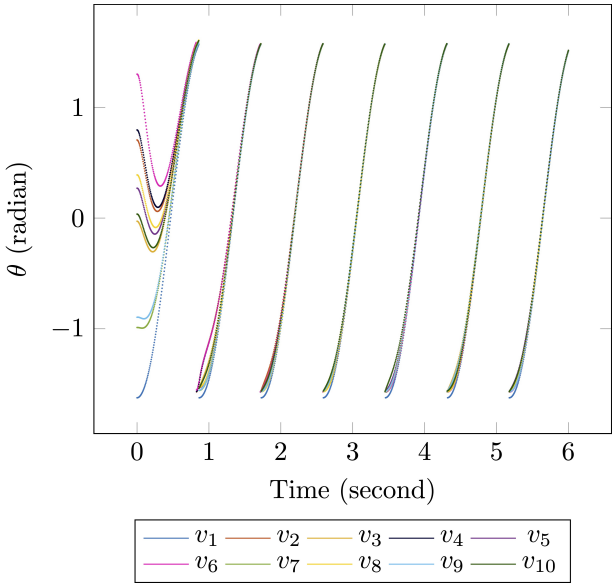

Fig. 5. Synchronization of robots communicating as shown in network Fig. 3 with controller (6) applied. All nodes have $\lambda_{1,i} = 6.5$ and $\lambda_{1,i} = 7$.

alone. This indicates that the model-dependent controllers perform better than the model-free controller. However, they require precise knowledge of the system parameters. Motivated to address this drawback, we will discuss a data-enabled method to estimate these system parameters in the next section.

8. PARAMETER ESTIMATION

The goal of this section is to discuss a tool for accurately estimating the model parameters ("lumped parameters") of the system. As a result, controllers that rely on accurate parameter values can still be a desirable option when applied with parameter estimation in real systems. For any node, this method can be carried out to estimate its parameters and does not rely on interactions between the

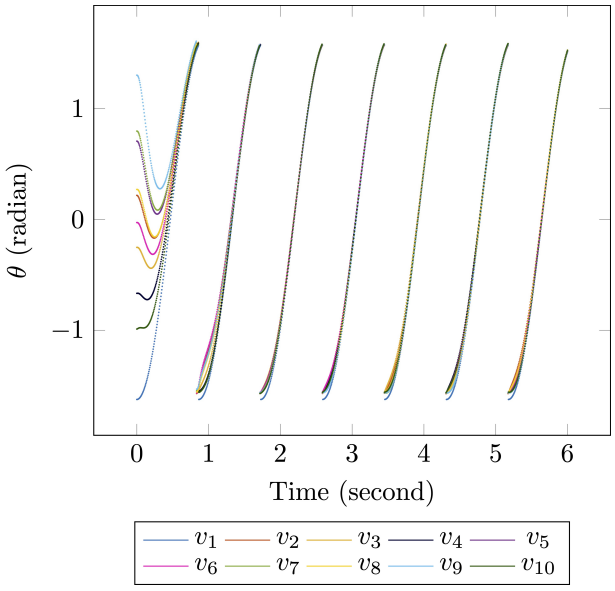

Fig. 6. Synchronization of robots communicating as shown in network Fig. 4 with controller (6) applied. All nodes have $\lambda_{1,i} = 6.5$ and $\lambda_{1,i} = 7$.

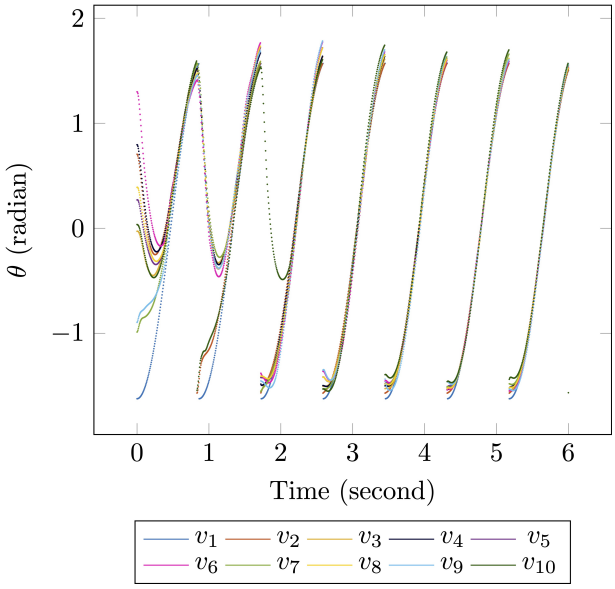

Fig. 7. Synchronization of robots communicating as shown in network Fig. 3 with controller (8) applied. All nodes have $K_i = [121, 22]^T$.

nodes. In light of the property that the Lagrange equation of motion of a multi-body system is affine in the lumped parameters, The dynamical system represented in (1) for a given brachiating robot can be rewritten as follows [36]:

$$Y(q,\dot{q},\ddot{q})p = \begin{bmatrix} 0 \\ \tau \end{bmatrix}, \tag{38}$$

where $Y(q,\dot{q},\ddot{q})$ is the data matrix, whose values are obtained from state measurement data, and defined in (39).

The task now is to estimate the value of the lumped parameters in (39) via an iterative approach. A brachiating robot is simulated for 5 seconds and the true values of lumped parameters used in the simulation are listed in the last column of the table 4. The input to the dynamics is a control signal $\tau = 5 + 0.1(4+t) + \sin(t) + \sin(0.5+3t)$. We record the discrete τ values and the state response to the input. The joint accelerations are computed by numerical derivative while simulating the system. Gaussian noise of magnitude 0.001 was added to the joint velocity and acceleration measurements from the simulation to emulate sensor noise.

We randomly sample N points from the outputs of the simulation and run the code presented in algorithm 1. The algorithm aims to find the set of values for $\hat{p}$, which is the estimate of p, that minimizes the following quadratic cost function:

$$\text{cost} = \frac{1}{2N}\sum_{k=1}^{k=N}\left\| Y_k\hat{p} - \begin{bmatrix} 0 \\ \tau_k \end{bmatrix}\right\|_2^2; \quad \text{with } lb \le \hat{p} \le ub. \tag{40}$$

The lower bounds lb and upper bounds ub for the cost function are defined as 50% below and 50% above the actual lumped parameter value. These bounds help the optimization algorithm with initial guesses close enough to the actual values to guarantee convergence.

The summary of the result is presented in Table 4, which shows very close estimations. Through our testing, it was observed that lower noise magnitudes result in closer estimation of the lumped parameters than the presented estimates.

Algorithm 1 Secure introspection

1: **while** iteration $\le$ maxIteration **do**
2: randomly select N points and compute cost from (40).
3: $value(iteration) = \arg\min_{\hat{p}}$ cost from (40).
4: $\hat{p} = \min(value)$

9. DISCUSSION AND CONCLUSION

This paper proposes control schemes to synchronize the motion of multiple underactuated robots in a network that leads to the synchronization of brachiation stride time independent of the network topology. Three controllers were developed for underactuated robot synchronization: one with partial feedback linearization, one with a linear–quadratic regulator, and one with sliding mode. The efficacy of these synchronization controllers is verified through a series of simulations with different network topologies. The proposed method is independent of the

Table 4. Summary of parameter estimation.

Lumped parameters	N = 10	N = 20	N = 30	N = 40	N = 50	True value
I_1	0.08999821	0.089999355	0.090000738	0.090000169	0.090001207	0.09
I_2	0.033000622	0.03299969	0.033000046	0.033000399	0.033000124	0.033
$m_1 r_1^2$	0.599827503	0.599725592	0.599728299	0.599753341	0.599805237	0.5997
$m_2 r_2^2$	0.136620952	0.136603307	0.136616683	0.136631339	0.136616522	0.1366
$g m_1 r_1$	14.20576946	14.20806744	14.21405206	14.17628303	14.21646778	14.2106
$g m_2 r_2$	4.023087837	4.025743909	4.024618111	4.023820674	4.023949987	4.0246
$l_1 m_2 r_2$	0.205138769	0.205131906	0.205111391	0.205149567	0.205166007	0.2051
$g l_1 m_2$	6.049503698	6.044481769	6.047888486	6.077792586	6.045614833	6.043
$m_2 l_1^2$	0.307971725	0.308003085	0.308007047	0.307968716	0.307958267	0.308

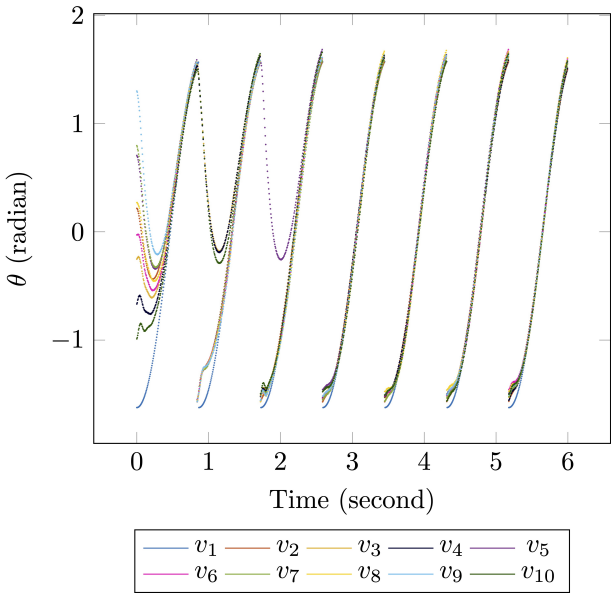

Fig. 8. Synchronization of robots communicating as shown in network Fig. 4 with controller (8) applied. All nodes have $K_i = [1000, 205]^T$.

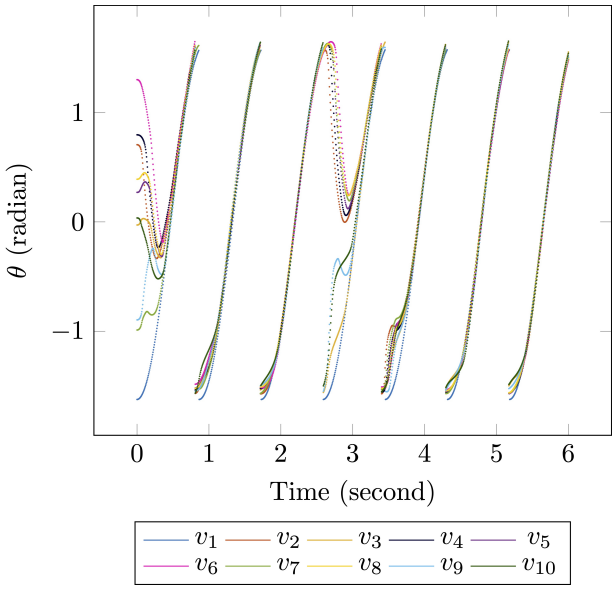

Fig. 9. Synchronization of robots communicating as shown in network Fig. 3 with controller (18) applied.

number of brachiating robots, creating a general framework that can be scaled. The paper also elaborates on a data-enabled algorithm for estimating the system model parameters, which can be used with model-dependent synchronization controllers. The proposed method is independent in terms of network topology and the number of brachiating robots. A future research direction would be to integrate machine learning techniques, e.g. reinforcement learning or recursive neural networks, with the proposed controllers to accomplish complicated coordination tasks with a network of underactuated systems.

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. National Science Foundation under CMMI Grant #2138206. Portion of the Minnesota State University Mankato Faculty Research Grant was also used to support this work.

CONFLICT OF INTEREST

The authors declare that there is no competing financial interest or personal relationship that could have appeared to influence the work reported in this paper.

REFERENCES

[1] W. Lin, Z. Qu, and M. A. Simaan, "Distributed game strategy design with application to multi-agent formation control," *Proc. of 53rd IEEE Conference on Decision and Control*, pp. 433–438, 2014.

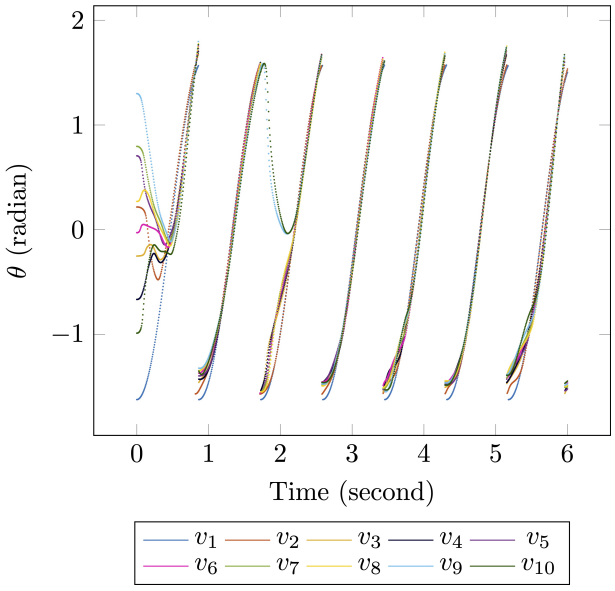

Fig. 10. Synchronization of robots communicating as shown in network Fig. 4 with controller (18) applied.

[2] Q. Van Tran, "Fixed-time localization of local coordinate frames: Interpretation and applications to formation control problems," *International Journal of Control, Automation and Systems*, pp. 1–10, 2023.

[3] X.-J. Wu, L. Xu, R. Zhen, and X.-L. Wu, "Global and local moth-flame optimization algorithm for uav formation path planning under multi-constraints," *International Journal of Control, Automation and Systems*, vol. 21, no. 3, pp. 1032–1047, 2023.

[4] H. Kim, D. Kim, H. Kim, J.-U. Shin, and H. Myung, "An extended any-angle path planning algorithm for maintaining formation of multi-agent jellyfish elimination robot system," *International Journal of Control, Automation and Systems*, vol. 14, no. 2, pp. 598–607, 2016.

[5] K.-K. Oh, M.-C. Park, and H.-S. Ahn, "A survey of multi-agent formation control," *Automatica*, vol. 53, pp. 424–440, 2015.

[6] K. Ozaki, H. Asama, Y. Ishida, A. Matsumoto, K. Yokota, H. Kaetsu, and I. Endo, "Synchronized motion by multiple mobile robots using communication," *Proc. of the International Conference on Intelligent Robots and Systems (IROS)*, vol. 2, pp. 1164–1169, 1993.

[7] N. Chopra and M. W. Spong, "On Exponential Synchronization of Kuramoto Oscillators," *IEEE Transactions on Automatic Control*, vol. 54, no. 2, pp. 353–357, 2009.

[8] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[9] C. Deng and G.-H. Yang, "Leaderless and leader-following consensus of linear multi-agent systems with distributed event-triggered estimators," *Journal of the Franklin Institute*, vol. 356, no. 1, pp. 309–333, 2019.

[10] Y. Yang and W. Hu, "Consensus of Double-Integrator Multi-Agent Systems with Directed Networks and Nonuniform Communication Time Delays," *Proc. of the 16th International Conference on Control Automation*, pp. 1229–1234, 2020.

[11] J. Jiang and Y. Jiang, "Leader-following consensus of linear time-varying multi-agent systems under fixed and switching topologies," *Automatica*, vol. 113, p. 108804, 2020.

[12] L. Chen, C. Liu, Z. Chu, A. M. Lopes, and Y. Chen, "Leader–follower weighted consensus of nonlinear fractional-order multiagent systems using current and time delay state information," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2024.

[13] Y. Bouteraa and J. Ghommam, "Synchronization control of multiple robots manipulators," *Proc. of the 6th International Multi-Conference on Systems, Signals and Devices*, pp. 1–6, 2009.

[14] S.-J. Chung and J.-J. E. Slotine, "Cooperative Robot Control and Concurrent Synchronization of Lagrangian Systems," *IEEE Transactions on Robotics*, vol. 25, no. 3, pp. 686–700, 2009.

[15] T. Tao, S. Roy, B. De Schutter, and S. Baldi, "Adaptive synchronization of uncertain underactuated euler-lagrange agents," *IEEE Transactions on Automatic Control*, 2024.

[16] K. D. Nguyen and H. Dankowicz, "Synchronization and consensus of a robot network on an underactuated dynamic platform," *Proc. of the International Conference on Intelligent Robots and Systems*, pp. 117–122, 2014.

[17] K.-D. Nguyen and H. Dankowicz, "Cooperative control of networked robots on a dynamic platform in the presence of communication delays," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 9, pp. 1433–1461, 2017.

[18] T. Chen, J. Shan, and G. Ramkumar, "Distributed adaptive control for multiple under-actuated Lagrangian systems under fixed or switching topology," *Nonlinear Dynamics*, vol. 93, no. 3, pp. 1705–1718, 2018.

[19] F. Dörfler and F. Bullo, "Synchronization in complex networks of phase oscillators: A survey," *Automatica*, vol. 50, no. 6, pp. 1539–1564, 2014.

[20] D. Sun, "Position synchronization of multiple motion axes with adaptive coupling control," *Automatica*, vol. 39, no. 6, pp. 997–1005, 2003.

[21] N. Chopra, M. W. Spong, and R. Lozano, "Synchronization of bilateral teleoperators with time delay," *Automatica*, vol. 44, no. 8, pp. 2142–2148, 2008.

[22] A. Rodriguez-Angeles and H. Nijmeijer, "Mutual synchronization of robots via estimated state feedback: a cooperative approach," *IEEE Transactions on Control Systems Technology*, vol. 12, no. 4, pp. 542–554, 2004.

[23] M. Javadi, D. Harnack, P. Stocco, S. Kumar, S. Vyas, D. Pizzutilo, and F. Kirchner, "Acromonk: A minimalist underactuated brachiating robot," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3637–3644, 2023.

$$Y = \begin{bmatrix} \ddot{q}_1 & \ddot{q}_1+\ddot{q}_2 & \ddot{q}_1 & \ddot{q}_1+\ddot{q}_2 & \sin(q_1) & \sin(q_1+q_2) & \cos(q_2)(2\ddot{q}_1+\ddot{q}_2)-\sin(q_2)\left(\dot{q}_2^2+2\dot{q}_1\dot{q}_2\right) & \sin(q_1) & \ddot{q}_1 \\ 0 & \ddot{q}_1+\ddot{q}_2 & 0 & \ddot{q}_1+\ddot{q}_2 & 0 & \sin(q_1+q_2) & \sin(q_2)\dot{q}_1^2+\ddot{q}_1 cos(q_2) & 0 & 0 \end{bmatrix}$$

$$P = \begin{bmatrix} I_1 & I_2 & m_1r_1^2 & m_2r_2^2 & gm_1r_1 & gm_2r_2 & l_1m_2r_2 & gl_1m_2 & m_2l_1^2 \end{bmatrix}^T \tag{39}$$

[24] S. Farzan, A.-P. Hu, M. Bick, and J. Rogers, "Robust Control Synthesis and Verification for Wire-Borne Underactuated Brachiating Robots Using Sum-of-Squares Optimization," *arXiv:2007.12047 [cs, eess]*, 2020.

[25] S. S. Grama, M. Javadi, S. Kumar, H. Z. Boroujeni, and F. Kirchner, "Ricmonk: A three-link brachiation robot with passive grippers for energy-efficient brachiation," *arXiv preprint arXiv:2403.15762*, 2024.

[26] X. Zhang, Z. Ji, H. Zhang, and R. Xiong, "A deep reinforcement learning control method for a four-link brachiation robot," in *2023 2nd International Conference on Machine Learning, Cloud Computing and Intelligent Mining (MLCCIM)*. IEEE, 2023, pp. 528–533.

[27] J. Nakanishi, T. Fukuda, and D. Koditschek, "A brachiating robot controller," *IEEE Transactions on Robotics and Automation*, vol. 16, no. 2, pp. 109–123, 2000.

[28] K.-D. Nguyen and D. Liu, "Gibbon-inspired Robust Asymmetric Brachiation along an Upward Slope," *International Journal of Control, Automation and Systems*, vol. 17, no. 10, pp. 2647–2654, 2019.

[29] J. Rodriguez-Maldonado, "Estimation of angular velocity and acceleration with kalman filter, based on position measurement only," *Measurement*, vol. 145, pp. 130–136, 2019.

[30] S. A. B. Birjandi, J. Kühn, and S. Haddadin, "Joint velocity and acceleration estimation in serial chain rigid body and flexible joint manipulators," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 7503–7509.

[31] M. Hamandi, M. Tognon, and A. Franchi, "Direct acceleration feedback control of quadrotor aerial vehicles," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 5335–5341.

[32] W. Shang and S. Cong, "Motion control of parallel manipulators using acceleration feedback," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 1, pp. 314–321, 2013.

[33] R. A. Hom and C. R. Johnson, "Matrix analysis," *Cambridge University Express*, vol. 455, 1985.

[34] J. Mei, W. Ren, and G. Ma, "Distributed containment control for Lagrangian networks with parametric uncertainties under a directed graph," *Automatica*, vol. 48, no. 4, pp. 653–659, 2012.

[35] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Transactions on Automatic Control*, vol. 49, no. 9, pp. 1520–1533, 2004.

[36] K.-D. Nguyen and H. Dankowicz, "Adaptive control of underactuated robots with unmodeled dynamics," *Robotics and Autonomous Systems*, vol. 64, pp. 84–99, 2015.

Praneel Acharya received the Ph.D. degree in mechanical enginnering from the South Dakota State University, Brookings, USA, in 2023. He is currently working as an Assistant Professor in Department of Aviation at Minnesota State University Mankato, MN. His research interests include manipulation for robotic systems, motion planning, control, and computer vision.

Kim-Doang Nguyen received the Ph.D. degree in mechanical engineering from the University of Illinois, Urbana-Champaign, Champaign, IL, USA, in 2015. He is currently an Assistant Professor in the Department of Mechanical and Civil Engineering with Florida Institute of Technology, Melbourne, FL, USA. His research interests include deep leaning, robotics, mechatronics, adaptive control, applied machine learning, and time-delay systems.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.